Information Maximizing Preference Learning

Yu Xuan Liu
Department of Electrical Engineering and Computer Science
University of California, Berkeley
yuxuanliu@berkeley.edu

Abstract

Specifying rewards for autonomous agents remains a challenging problem. While preference learning has been shown to be effective for reward learning, it's important to make use of human labels efficiently. In this paper, we propose a method for querying human labels that maximizes information gain of the reward function. In some settings we show that our method can make efficient use of human labels compared to baselines methods.

1 Introduction

Reward specification has been a long-standing problem in reinforcement learning. Designing well-behaved reward functions is a challenging process that must account for a number of factors such as exploitation and sparsity. A poorly-specified reward may cause the agent to perform behaviors that were unintended. For instance, a self-driving car that's rewarded to drive closer in Euclidean distance to the destination may learn to plow through pedestrians and drive off roads in the direction of the destination. Moreover, a reward that is too sparse requires a significant amount of experience before the agent can achieve any learning signal.

Recent work by Christiano et al. (2017) proposes a method to learn reward functions directly from observations using human preference queries. In this work, a human is presented with two videos of agent performance and must select the "better" performing one for a given task. A reward function is then fit for all labels provided by the human, and the agent optimizes the reward to produce more videos for the human to label.

However, the method for selecting videos for human labelling uses a heuristic for uncertainty which can sometimes perform worse than random selection. Requiring a large number of human labels can also be expensive and time-consuming. A method that can query labels efficiently would greatly improve the effectiveness of preference learning methods.

In this paper, we explore an information theoretic approach for querying human preferences. We start by describing the problem setup and related work. Then we describe our method and the algorithm details. Finally we will evaluate our method in a number of experiments and discuss the results.

2 Related Work

We follow the same algorithmic framework of Christiano et al. (2017), which interleaves preference learning, reinforcement learning, and labelling. In this method, a predicted reward function $\hat{r}$ is learned from human preferences. An agent optimizes the predicted reward $\hat{r}$ using Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) and produces observation sequences for labelling. A human annotates the sequences, and the predicted reward function is then trained to match the labelled preferences. The agent optimization and preference annotation occur concurrently until termination.

Preprint. Work in progress.

Blundell et al. (2015) explored a model for Bayesian neural networks (BNN) in which the network is parameterized by a random variable θ. This formulation provides a model for uncertainty in the network weights. Houthooft et al. (2016) introduced a method for estimating information gain using BNNs. They approximate the dynamics of the environment using a BNN and reward the agent for states that maximize information gain of the dynamics estimate. This information gain reward is a natural exploration bonus that enables the agent to effectively explore the environment.

3 Problem Formulation

In our problem setting, an agent acts in the environment to optimize an unknown reward function. Our environments can be modeled as Markov decision processes with initial state distribution $s_0 \sim p(s_0)$, transition function $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$, and observation mapping $o_t \sim p(o_t|s_t)$. Each environment has a true reward function $r(s_t, a_t)$ which defines the task for that environment.

To learn the true reward function, the agent can select pairs of trajectory segments to query information about the true reward from a labelling source. The labelling source is given two sequences of observations, $\sigma^1 = \{o_1^1, ..., o_n^1\}$, $\sigma^2 = \{o_1^2, ..., o_n^2\}$ and produces a distribution $\mu(\sigma)$ determining which observation sequence is preferred. Three labelling choices are available ranging from $\mu(\sigma^1) = 1$ for preferring σ^1, $\mu(\sigma^2) = 1$ for preferring σ^2, and $\mu(\sigma^1) = \mu(\sigma^2) = 0.5$ for indifference.

We evaluate preference learning on two main sources of labels. The first is synthetic labels in which the true environment reward is used to determine preferences. In this setting $\sigma^1 \succ \sigma^2$ if $\sum_i^n r(o_i^1, a_i^1) > \sum_i^n r(o_i^2, a_i^2)$ and vice versa, with indifference when the sum of rewards are equal. The second setting is human labels in which a human is presented with a web interface that displays loops of two observation sequences. The human must indicate preference with "left is better", "right is better", "tie" for indifference, or "can't tell" to discard the sample.

The objective of the agent is to find a policy $\pi(s|a)$ that optimizes the expected true reward $E_\pi[r(s, a)]$. The agent can take actions in the environment and send queries for preferences under the true reward.

4 Model

4.1 Preference learning

As in Christiano et al. (2017) we choose to model the probability of a human preferring a sequence to be exponential in the sum of predicted rewards of the sequence:

$$P(\sigma^1 \succ \sigma^2) = \frac{\exp\{\sum_i^n \hat{r}(o_i^1, a_i^1)\}}{\exp\{\sum_i^n \hat{r}(o_i^1, a_i^1)\} + \exp\{\sum_i^n \hat{r}(o_i^2, a_i^2)\}}$$

The predicted reward function is then learned using cross entropy loss:

$$L(\hat{r}) = -\sum_{\sigma^1, \sigma^2, \mu} (\mu(\sigma^1) P(\sigma^1 \succ \sigma^2) + \mu(\sigma^2) P(\sigma^2 \succ \sigma^1))$$

4.2 Querying schemes

It is natural that certain pairs of observation sequences will be more informative for preference learning. A method that can select such pairs would offer a reduction in annotation costs and time. Prior work considered two schemes for select which observations to label. The first is random in which random pairs of observation sequences are generated by the agent and selected for labelling. The second uses an ensemble of reward predictors $\hat{r}$, and estimates uncertainty in the predictions by selecting trajectory pairs with the highest prediction variance across the ensemble.

While these querying schemes can be used to learn reasonable preferences, it is not clear that they are the best or most efficient methods. Indeed, prior work has shown that the random sampling method can outperform the ensemble method in some tasks.

A more natural objective would be selecting queries that maximize information gain with respect to the reward estimate $\hat{r}$:

$$I(\hat{r}; \sigma^1, \sigma^2, \mu | D)$$

Algorithm 1 Information maximizing preference learning

for iteration $i \in \{1, \ldots, N\}$ **do**
  Sample k trajectories $\{\tau_1, ..., \tau_k\}$ from the policy π and environment
  Update π using TRPO with reward $\hat{r}_\theta$ where $\theta \sim q(\theta; \phi)$
  if more human labels should be collected according to the labelling schedule **then**
    Sample observations $\sigma^1, ..., \sigma^m$ from the policy π
    Rank pair of observations using the information estimate: $\mathbb{E}_\mu[D_{KL}[q(\theta|\phi')||p(\theta|\phi)]]$
    Select top t observation pairs for human labelling, add labels to dataset D
    Train $q(\theta; \phi)$ by maximizing variational lower bound: $\mathbb{E}_q[\log p(D|\theta)] - D_{KL}[q(\theta; \phi)||p(\theta)]$
    Update $p_\psi(\mu|\sigma^1, \sigma^2)$ with maximum likelihood on D
  end if
end for

Where D is the dataset of labels collected so far. While this objective is generally intractable for neural network reward estimators, we use the machinery for estimating information gain as described in Houthooft et al. (2016). In this formulation, the reward estimator $\hat{r}_\theta$ is a Bayesian neural network (BNN) (Blundell et al., 2015) parameterized by a random variable θ with prior distribution $p(\theta)$. Writing the objective in terms of θ, we arrive at:

$$I(\theta; \sigma^1, \sigma^2, \mu|D) = \mathbb{E}_{\sigma^1,\sigma^2,\mu}[D_{KL}[p(\theta|\sigma^1, \sigma^2, \mu, D)||p(\theta|D)]]$$

We approximate $p(\theta|D)$ with $q(\theta; \phi)$, using variational inference by maximizing the variational lower bound:

$$\mathbb{E}_q[\log p(D|\theta)] - D_{KL}[q(\theta; \phi)||p(\theta)]$$

Then we can approximate the information in terms of $q(\theta; \phi)$:

$$\mathbb{E}_{\sigma^1,\sigma^2,\mu}[D_{KL}[q(\theta|\phi')||p(\theta|\phi)]]$$

Where ϕ' are the updated parameters of the belief over the reward function after observing the label σ^1, σ^2, μ. Intuitively, we can see this as how much the belief over the reward function changes after observing a label; a label that is informative will change our belief of the reward function more than an uninformative label. We direct the reader to Houthooft et al. (2016) for more details regarding this information estimate.

In maximizing this information measure, we must select distributions for σ^1 and σ^2. One possibility is to incentivize the agent's policy π to take sequences of actions that are more informative for labelling. While this is a promising approach, we leave it for future work. Rather, we still assume that observation sequences $\sigma^1 \sim \pi$ and $\sigma^2 \sim \pi$ are sampled from the agent's current policy π. Then, we rank pair of sequences based off of the information measure $\mathbb{E}_\mu[D_{KL}[q(\theta|\phi')||p(\theta|\phi)]]$. To do this we must estimate $p_\psi(\mu|\sigma^1, \sigma^2)$ which we parameterize with ψ and learn using maximum likelihood. The full details of the method are outlined in Algorithm 1.

5 Experiments

5.1 Environments

We will consider a number of environments in OpenAI Gym (Brockman et al., 2016) as well as a simple environment for visualizing different aspects of our method.

The simple environment we consider is a 2D navigation setting in which a point mass must navigate to a goal in a bounded, 6×6 box. The agent is initialized in the center of the box and can move horizontally and vertically by any amount within the range $[-0.1, 0.1]$. A goal is located at a fixed point in the environment and the agent is rewarded $r(s_t, a_t) = -||s_t - g||_2$ where s_t is the agent location and g is the goal location. The agent observes its location $o_t = s_t$ as a two dimensional vector.

We also evaluate our system on OpenAI Gym environments, most of which are based on the Mujoco simulator. We consider three environments Hopper, Reacher, and Pendulum as show in Figure 1. In

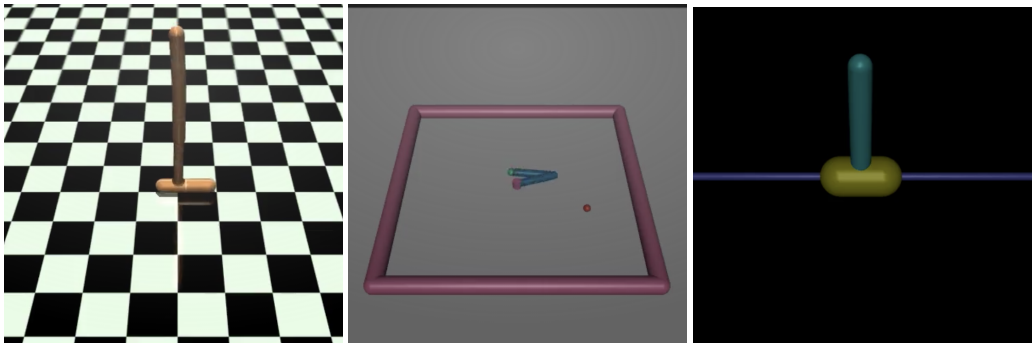

Figure 1: Visualization of the OpenAI Gym environments. From left to right: Hopper, Reacher, Pendulum.

the Hopper environment, a torque-controlled, planar leg must move to the right of the environment. The hopper has three torque-actuated joints on its thigh, leg, and foot. The observations consist of the positions and velocities of the hopper and its joints angles. In Reacher, a 2-DoF, torque-controlled arm must move its end-effector to a given goal. The reacher has two torque-actuated joints at its base and arm. The observations consist of the positions and velocities of the reacher and its joints angles as well as the relative position to the goal. In Pendulum, a torque-controlled pendulum is rewarded for keeping upright. There is one torque-actuated slider at the bottom, and the observations consist of the slide position and the angle of the pendulum.

5.2 Experiments

In evaluating our method, we sought to address the following questions:

1. Does information maximizing preference learning improve the sample efficiency of preference labelling?
2. How do the reward landscapes learned by our method compare to prior work?
3. Does the information metric produce rankings of observation sequences that are reasonable?

To evaluate sample efficiency, we compare our method with the baseline random querying method. We evaluate on the Open AI gym environments and the 2D navigation environment as described in Section 5.1. For different amounts of human labelled we plot the learning curves of the average true reward for each method. Please see the attached videos for visualizations of the agent performance over the training procedure.

In the baseline methods, we parameterize the reward estimate $\hat{r}$ using a two-layer, 64-unit, fully-connected neural network with Relu activations. We train using the Adam optimizer with learning rate 10^{-3}. In our method, we parameterize the reward estimate using a two-layer, fully-connected BNN, with 64-units each. We use Relu activation and optimize using the Adam optimizer with

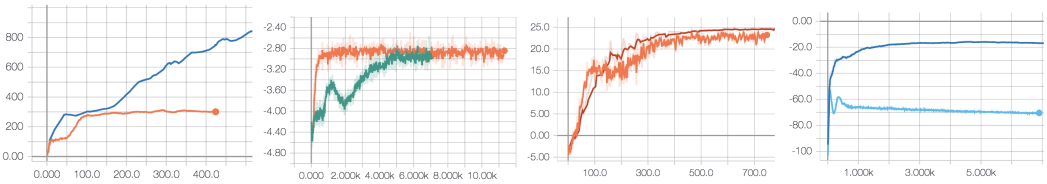

Figure 2: Comparison of our method with the baseline random query method. We find that our method is comparable to the baseline in Reacher and Pendulum, better than the baseline on 2D navigation, and worse on Hopper. From left to right: comparison of our method and the baseline on Hopper (ours: orange, baseline: blue), Reacher (ours: green, baseline: orange), Pendulum (ours: orange, baseline: red), and 2D navigation (ours: dark blue, baseline: light blue).

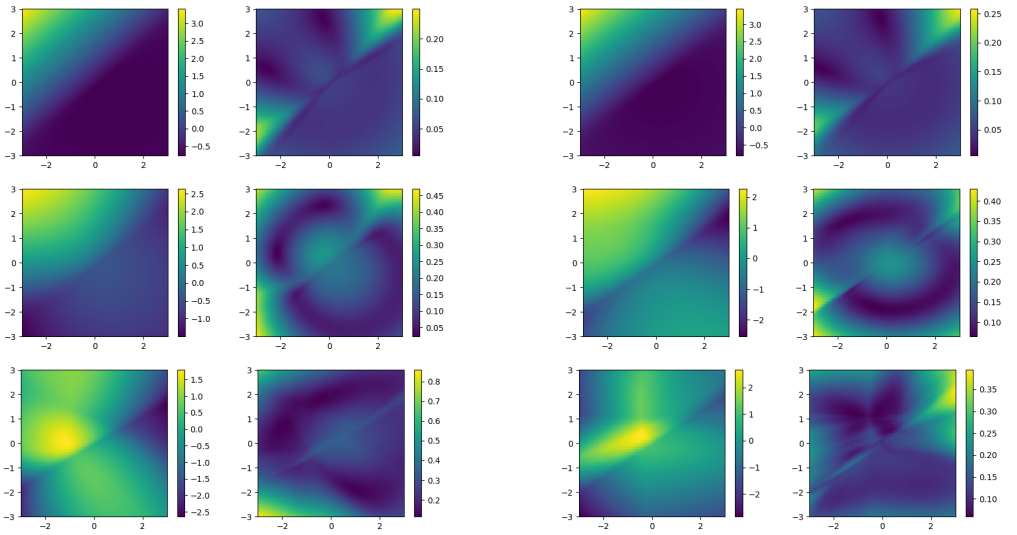

Figure 3: Predicted reward landscapes for the 2D navigation task using our method with a goal situated at (-1.5, 1.5). Each of the two plots shows both the mean (left) and standard deviation (right) computed across multiple samples of BNN parameter distribution. From left to right and top to bottom: we plot the reward landscape after training on 100, 500, 1000, 5000, 10000, 40000 simulated trajectories.

learning rate 10^{-3}. We use a BNN weight prior of $p(\theta) = \mathcal{N}(0, 0.2)$. Instead of parameterizing ϕ as a separate network, we assume $P(\sigma^1 \succ \sigma^2)$ is sufficient in estimating the user decision.

The results are summarized in Figure 2. We found that our method performed comparable to the baseline in two environments, worse in one, and better in another. While our method did not achieve better performance across all environments, there is promise in better performance with a more comprehensive hyperparameter sweep.

Next we evaluate reward landscapes on the 2D navigation environment since it is sufficiently low dimensional to visualize. In this environment, the reward is the negative Euclidean distance from the goal. We visualize the reward landscapes learned throughout the training procedure for our method.

Since the BNN is a stochastic neural network, we compute the reward landscape over multiple samples of the network parameters. To consolidate the different reward scales, we normalize the estimated rewards to be 0 mean and standard deviation 1. Then we compute the mean and standard deviation of the reward landscape over multiple parameter samples. As shown in Figure 3. we find that the network initially makes an overgeneralization by rewarding the top left region. Then over time, the network begins to decrease its estimate of the top left reward and shift towards the true goal.

To understand if the information metric is meaningful, we show various pair of observations ranked by the information metric. To understand the state of the reward estimate, we also plot the reward landscape. All plots are computed on the 2D navigation environment for easier visualization.

As shown in Figure 4, we see that the information metric makes reasonable predictions of which trajectory segments will be most informative. When the BNN is uncertain about the reward at the top and left regions, information about those regions would be most informative. Meanwhile, trajectories that are very similar are unlikely to offer much information on the reward.

6 Code

We will base our code off of the open source implementations of Christiano et al. (2017) and Houthooft et al. (2016). We started by verifying that the existing implementations for both methods work as expected. Then we set up simple environments where visualizing learned reward functions and agent performance is easy for debugging. Then we set up the models and ensure that they are

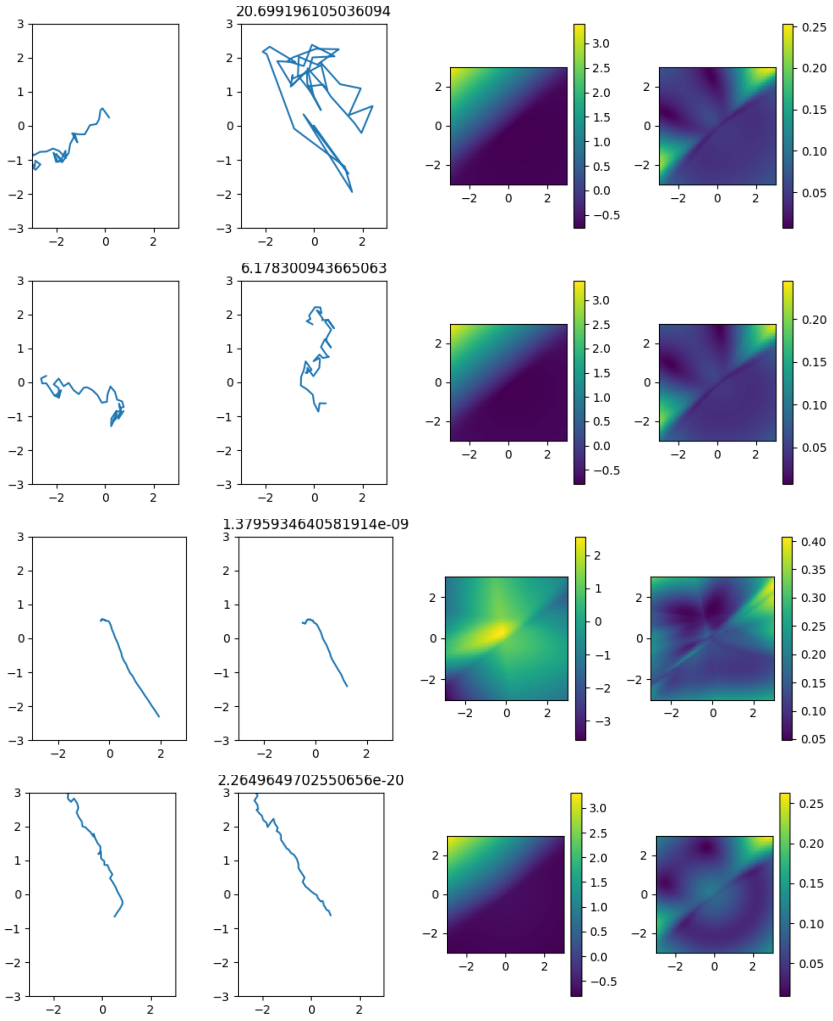

Figure 4: Four plots of the information metric in decreasing order from top to bottom. In each plot, we show σ^1 (left), σ^2 (left-middle), the reward mean (right-middle), and reward standard deviation (right). The title above the second plot is the estimated information gain (in nats) if a human were to label it. We find that the information metric provides a reasonable estimate given the reward landscape: trajectories spanning regions of uncertainty are more likely to have higher information gain than similar trajectories in well-understood regions of the reward.

performing as expected before integrating the full method together. We evaluate our method on simple tasks to sanity check the performance and then scale to more complicated tasks.

We modified the BNN implementation in Houthooft et al. (2016) to estimate human perference instead of environment dynamics. Then we tested the BNN as a reward estimate without using any information estimate. We added visualization of the BNN and implemented the information metric for querying.

7 Discussion

In working on this project, it has been difficult reconciling two codebases for fairly different projects. Moreover, this project has a number of different components. Verifying that each component works independently and then combining them to work together was challenging. Training the BNN was tricky as the prior was initially too large causing the predictions to saturate and training to diverge.

After reducing the prior, the BNN became much easier to train. A larger hyperparameter sweep could also lead to more improvements over prior methods.

There are a number of assumptions made by this method, for instance modeling human preference as exponential in the sum of rewards. The method also makes number of approximations due to the intractability of the objective for high-capacity neural network models. It would be interesting to develop more tractable formulations of the problem that do not suffer from the same approximation errors.

References

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 1613–1622. JMLR.org, 2015. URL `http://dl.acm.org/citation.cfm?id=3045118.3045290`.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. 06 2017.

Rein Houthooft, Xi Chen, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. In D. D. Lee, U. V. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances In Neural Information Processing Systems 29*, pp. 1109–1117. Curran Associates, Inc., 2016. URL `http://papers.nips.cc/paper/6591-vime-variational-information-maximizing-exploration.pdf`.

John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML*, 2015.